FAKULTAS ILMU KOMPUTER
UNIVERSITAS INDONESIA

Tutorial 5
CSIM603026 – Arsitektur dan Pemrograman Aplikasi Perusahaan
Semester Ganjil 2017/2018

Menggunakan Database serta Relasi Database dalam Project Spring Boot

Versi: 1, 2 Oktober 2017

Requirement & Installation

Pada tutorial kali ini Anda akan menggunakan dua library eksternal seperti pada tutorial sebelumnya, yaitu MyBatis dan Lombok. Sebelum menjalankan program Anda, pastikan Lombok dan MyBatis sudah ter-install lalu jalankan XAMPP atau database server Anda. Anda dapat menggunakan kembali database yang sudah dibuat pada tutorial sebelumnya atau anda dapat membuat sebuah database baru pada MySQL Anda (Disarankan menggunakan database sebelumnya). Pada tutorial sebelumnya Anda sudah membuat tabel **Student**. Sekarang kita akan melengkapi basis data dengan menambahkan tabel **Course**. Sehingga pada database terdapat tabel **Student** dan **Course** dengan spesifikasi seperti berikut:

STUDENT
Kolom :

- npm
  varchar(20), varchar 20, NOT NULL
- name
  varchar(45), DEFAULT NULL
- gpa
  double, DEFAULT NULL

Tambahkan kolom **npm** sebagai PRIMARY KEY.

COURSE
Kolom:

- id_course
  varchar(20), varchar 20, NOT NULL
- name
  varchar(45), DEFAULT NULL
- credits
  double, DEFAULT NULL

Tambahkan kolom **id_course** sebagai PRIMARY KEY.

POPULASIKAN TABEL

Contoh populasi tabel **course** (diperbolehkan menambahkan atau mempopulasikan dengan data lain). Silakan gunakan script dibawah atau tambahkan secara manual melalui phpMyAdmin.

```
INSERT INTO `course` (`id_course`, `name`, `credits`) VALUES
('CSC123', 'PSP', 4),
('CSC124', 'SDA', 3),
('CSC125', 'DDP 1', 4),
('CSC126', 'MPKT', 6);
```

Penjelasan

Dalam tutorial ini kita akan menggunakan kembali **project dari tutorial sebelumnya**. Anda perlu melakukan **copy project tutorial 04 yang sudah dikerjakan minggu lalu**.

Anda perlu memastikan apakah Lombok, MyBatis, MySQL sudah terinstall atau belum dengan membuka pom.xml dan cek apakah ketika dependency tersebut telah tertulis diantara <dependencies></dependencies>

Sebelumnya kita sudah mencoba basis data dengan Spring Boot menggunakan MyBatis. Namun, kita hanya menggunakan 1 entity yaitu entity **Student**. Pada tutorial ini kita akan melanjutkan tutorial 04 dengan menambahkan entity yang baru yaitu **Course** dan menghubungkannya dengan **Student**.

ERD untuk hubungan antara Course dan Student adalah sebagai berikut:

Hubungan antara Student dan Course adalah Many-to-Many karena Student dapat mengambil beberapa Course dan Course dapat diambil oleh beberapa Student.

Dikarenakan terdapat relasi Student take Course, maka perlu dibuat satu tabel baru yang disebut dengan **studentcourse** yang menyimpan relasi tersebut. Tabel tersebut akan mencatat **Student** mengambil **Course** apa saja. Tabel akan terdiri dari 2 kolom yaitu **npm mahasiswa dan id course**. Gunakan script SQL berikut:

```
CREATE TABLE IF NOT EXISTS `studentcourse` (
  `npm` varchar(20) NOT NULL,
  `id_course` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
INSERT INTO `studentcourse` (`npm`, `id_course`) VALUES
('123', 'CSC126'),
('124', 'CSC123'),
('124', 'CSC124');
```

Atau Anda dapat menambahkan secara manual melalui phpMyAdmin.

Pada tabel di atas diasumsikan terdapat mahasiswa dengan NPM 123 mengambil mata kuliah CSC126 dan NPM 124 mengambil mata kuliah PSP dan SDA.

Silakan populate database sesuai dengan data Student yang ada pada data Anda.

MENAMBAHKAN MODEL

1. Membuka kode tutorial 04 yang sudah dikerjakan minggu lalu.
2. Karena kita menambahkan Model baru maka kita perlu menambahkan class **CourseModel** pada package com.example.model

```
package com.example.model;

import java.util.List;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@AllArgsConstructor
@NoArgsConstructor
public class CourseModel {
        private String idCourse;
        private String name;
        private Integer credits;
        private List<StudentModel> students;
}
```

3. Karena Student memiliki daftar Course yang dia ambil, maka pada class **StudentModel** perlu ditambahkan list of CourseModel pada variabel:

```
...
private String npm;
private String name;
private Double gpa;
private List<CourseModel> courses;

...
```

Kemudian, pada **StudentController** akan muncul error karena menambahkan parameter constructor baru. Ubah inisialisasi constructor menjadi:

```
StudentModel student = new StudentModel (npm, name, gpa, null);
```

4. Selanjutnya kita akan membuat method pada **StudentMapper** yang mengembalikan list of course pada Student dengan NPM tertentu. Methodnya adalah sebagai berikut:

```
@Select("select course.id_course, name, credits " +
        "from studentcourse join course " +
        "on studentcourse.id_course = course.id_course " +
        "where studentcourse.npm = #{npm}")
List<CourseModel> selectCourses (@Param("npm") String npm);
```

Query tersebut akan melakukan join antara tabel studentcourse dengan course berdasarkan id_course dan difilter hanya pada student dengan NPM yang diberikan.

5. Selanjutnya kita akan menghubungkan method selectStudent dengan method selectCourses agar saat melakukan select maka variabel List<Course> juga akan terisi. Pada class **StudentMapper** terdapat method

```
@Select("select npm, name, gpa from student where npm = #{npm}")
StudentModel selectStudent (@Param("npm") String npm);
```

Method tersebut diubah menjadi

```
@Select("select npm, name, gpa from student where npm = #{npm}")
@Results(value = {
   @Result(property="npm", column="npm"),
   @Result(property="name", column="name"),
   @Result(property="gpa", column="gpa"),
   @Result(property="courses", column="npm",
           javaType = List.class,
           many=@Many(select="selectCourses"))
})
StudentModel selectStudent (@Param("npm") String npm);
```

Anotasi Results digunakan untuk memetakan hasil dari query select ke class model.

```
@Result(property="npm", column="npm")
```

Variabel **property** diisi dengan nama variabel di class StudentModel, sedangkan variabel **column** diisi dengan nama kolom hasil kueri di database (parameter yang dibutuhkan).

Sedangkan untuk variabel courses karena hasilnya diambil dari method lain maka querynya adalah sebagai berikut:

```
@Result(property="courses", column="npm",
        javaType = List.class,
        many=@Many(select="selectCourses"))
```

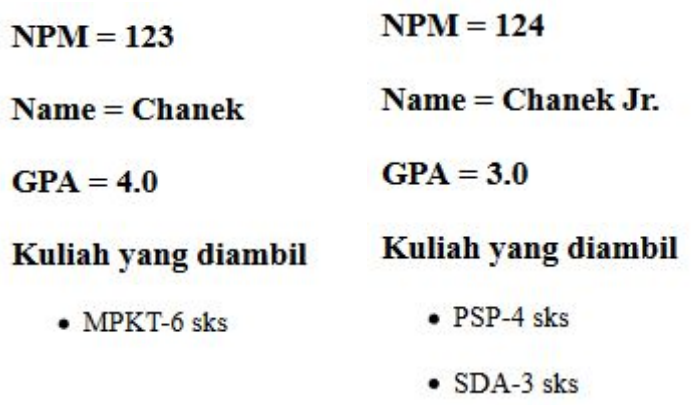
Property diset dengan variabel courses karena kita ingin mengisi variabel courses di class StudentModel. Variabel column diisi dengan npm karena kolom npm pada database akan dikirim menjadi parameter di method selectCourses. Variabel javaType menentukan class yang menjadi kembalian. Kemudian variabel many diset dengan method yang akan mengisi variabel courses yaitu selectCourses.

6. Selanjutnya kita akan menambahkan *view* pada **view.html** untuk menampilkan list kuliah yang diambil oleh mahasiswa seperti berikut:

```
<h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
<h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
<h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>

<h3>Kuliah yang diambil</h3>
<ul th:each="course,iterationStatus: ${student.courses}">
   <li th:text="${course.name} + '-' + ${course.credits} + ' sks'" >
       Nama kuliah-X SKS
    </li>
</ul>
```

7. Jalankan program tersebut dan buka http://localhost:8080/student/view/123 dan http://localhost:8080/student/view/124 (nomor student disesuaikan dengan database masing-masing).
8. Tampilannya kurang lebih sebagai berikut:



LATIHAN

1. Ubah method **selectAllStudents** pada kelas **StudentMapper** agar halaman *viewall* menampilkan semua *student* beserta daftar kuliah yang diambil.
   Contoh tampilan:

2. Buatlah view pada halaman **http://localhost:8080/course/view/{id}** untuk Course sehingga dapat menampilkan data course beserta Student yang mengambil.

ID = CSC123

Nama = PSP

SKS = 4

Mahasiswa yang mengambil

- 124 - Chanek Jr.

ID = CSC126

Nama = MPKT

SKS = 6

Mahasiswa yang mengambil

- 123 - Chanek

Deliverables

Deliverables untuk tutorial kali ini adalah:

1. **File Project**
   Buat sebuah project baru pada organization /apap-2017 dengan format nama tutorial5_NPM, contoh tutorial5_1501234567. Push project Anda ke repository GitHub tersebut.

2. **Write-up**
   Buat sebuah file write-up. Jelaskan apa saja hal yang Anda pelajari dari tutorial ini. Cantumkan juga penjelasan Anda terhadap hal-hal barikut:
   - Method yang Anda ubah pada Latihan Merubah **SelectAllStudents**, jelaskan
   - Method yang Anda buat pada Latihan Menambahkan **View** pada **Course**, jelaskan

   Kami rekomendasikan Anda menggunakan format pdf agar dapat lebih leluasa dalam menuangkan penjelasan Anda dengan dukungan screenshot dan kode. Masukkan file writeup ke folder project. Pastikan file write-up juga di-push ke repository.

Deadline

07 Oktober 2017, 23:59:59

Penalti

Penalti:

- Keterlambatan
  Penalti keterlambatan sebesar -10 poin akan ditambahkan setiap 10 menit keterlambatan